## K.2 A Survey of RISC Architectures for Desktop, Server, and Embedded Computers

### Notation | Meaning | Example | Meaning
--- | --- | --- | ---
<-- | Data transfer. Length of transfer is given by the destination’s length; the length is specified when not clear. | Regs[R1]<--Regs[R2]; | Transfer contents of R2 to R1. Registers have a fixed length, so transfers shorter than the register size must indicate which bits are used.
M | Array of memory accessed in bytes. The starting address for a transfer is indicated as the index to the memory array. | Regs[R1]<--M[x]; | Place contents of memory location x into R1. If a transfer starts at M[1] and requires 4 bytes, the transferred bytes are M[1], M[1+1], M[1+2], and M[1+3].
<--n | Transfer an n-bit field, used whenever length of transfer is not clear. | M[y]<--16M[x]; | Transfer 16 bits starting at memory location x to memory location y. The length of the two sides should match.
Xn | Subscript selects a bit. | Regs[R1]0<--0; | Change sign bit of R1 to 0. (Bits are numbered from MSB starting at 0.)
Xm..n | Subscript selects a field. | Regs[R3]24..31<--M[x]; | Moves contents of memory location x into low-order byte of R3.
X^n | Superscript replicates a bit field. | Regs[R3]0..23<--024; | Sets high-order 3 bytes of R3 to 0.
## | Concatenates two fields. | Regs[R3]<--024##M[x]; F2##F3<--64M[x]; | Moves contents of location x into low byte of R3; clears upper 3 bytes. Moves 64 bits from memory starting at location x; 1st 32 bits go into F2, 2nd 32 into F3.
*, & | Dereference a pointer; get the address of a variable. | p*<--&x; | Assign to object pointed to by p the address of the variable x.
<<, >> | C logical shifts (left, right). | Regs[R1] << 5 | Shift R1 left 5 bits.
==, !=, >, <, >=, <= | C relational operators; equal, not equal, greater, less, greater or equal, less or equal. | (Regs[R1]==Regs[R2]) & (Regs[R3]!=Regs[R4]) | True if contents of R1 equal the contents of R2 and contents of R3 do not equal the contents of R4.
&, |, ^, ! | C bitwise logical operations: AND, OR, XOR, and complement. | (Regs[R1] & (Regs[R2] | Regs[R3])); | Bitwise AND of R1 and bitwise OR of R2 and R3.

*Figure K.29* Hardware description notation (and some standard C operators).

<table>
<thead>
<tr>
<th>Found in architectures</th>
<th>(Plain) branch</th>
<th>Delayed branch</th>
<th>Annulling delayed branch</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alpha, PowerPC, ARM, Thumb, SuperH, M32R, MIPS16</td>
<td>MIPS64, PA-RISC, SPARC, SuperH</td>
<td>MIPS64, SPARC</td>
<td>PA-RISC</td>
</tr>
</tbody>
</table>

*Figure K.30* When the instruction following the branch is executed for three types of branches.